(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 283 477 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
12.02.2003 Bulletin 2003/07

(51) Int Cl.7: G06F 17/30

(21) Application number: 01119078.2

(22) Date of filing: 07.08.2001

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE TR
Designated Extension States:
AL LT LV MK RO SI

(71) Applicant: SAP Aktiengesellschaft
69190 Walldorf (DE)

(72) Inventors:
- Pfeifer, Marcus
  69117 Heidelberg (DE)
- Brinkmoeller, Bernhard
  69168 Wiesloch (DE)

(54) Method and computer system for identifying objects for archiving

(57) Method and computer system for identifying objects for archiving out of a plurality of objects. A first computer system supplies a plurality of objects for archiving. Each object has an object type (130) and has at least one sub-object. Further, an assignment scheme (190) is provided, which assigns a plurality of sub-object types (140, 141, 142) to the object type (130). A computer program identifies at least one sub-object for archiving with the object by using the assignment scheme (190). A second computer system receives data for archiving from the first computer system. The data comprise the object and the at least one sub-object that are identified by the first computer system for archiving. The second computer system stores the data.

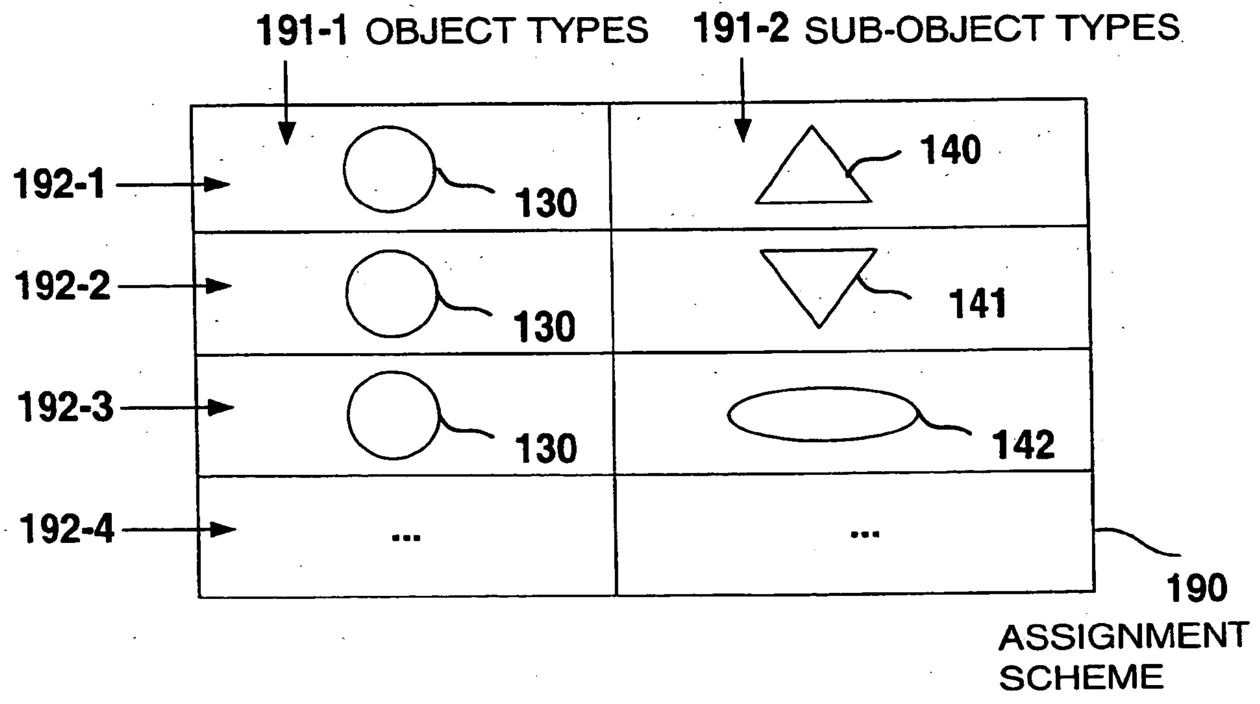

FIG. 5

Description

Field of the Invention

[0001] The present invention generally relates to electronic data processing, and more particularly, relates to method, computer program product and system for data archiving.

Background of the Invention

[0002] In prior art systems, such as Enterprise Resource Planning (ERP) systems (e.g. SAP R/3 from SAP AG) the archiving of data is achieved by providing an archiving program for each different type of data. Typically, the data that are subject to archiving are documents, such as a financial document (e.g. an accounting voucher) or a purchasing document, where items purchased from a certain vendor are listed. Another example of data that can be a subject to archiving is a database table.

[0003] In the latest prior art systems (e.g. SAP Customer Relationship Management) documents are objects that comprise sub-objects. Each object has a defined object type (e.g. purchasing document, invoice, accounting voucher, etc.). Each sub-object has a defined sub-object type (e.g. business partner, note, product, etc.). An object type includes a specific subset of sub-object types. For example, a purchasing document object comprises sub-objects of type business partner (e.g. address data of a supplier) and of type product (e.g. the line items of the purchasing document). An accounting voucher object would not comprise a product sub-object, because product information has no relevance in financial accounting. However, the accounting voucher object could comprise a business partner sub-object with the bank account of a supplier. Therefore, an assignment scheme defines assignments of sub-object types to object types. One sub-object type can be assigned to multiple object types. New object types can thus be created by composing a new subset of sub-object types.

[0004] Typically, each object/sub-object type has a unique data structure and the corresponding archiving program has to reflect this unique data structure to archive all data within the object. Whenever the data structure of an object/sub-object type is modified (e.g. a table is added to or removed from an object/sub-object), the corresponding archiving program does not take into account the modifications unless it is manually adjusted to the new data structure of the modified object/sub-object type. Without this adjustment some data within an object/sub-object of the modified object/sub-object type are not subject to archiving. Complex application systems, such as ERP systems, support a large number of different object/sub-object types, which makes it difficult to keep the data structures of the object/sub-object types consistent with the corresponding archiving programs.

Summary of the Invention

[0005] Hence, the present invention provides computer-implemented method, computer program product and computer system to solve the technical problem of inconsistencies between the data structure of objects/sub-objects that are subject to archiving and the corresponding archiving programs.

[0006] According to a first preferred embodiment of the present invention, the solution to the technical problem is provided by the following characteristics:

A first inventive computer-implemented method on an application computer for identifying objects (e.g. application objects) for archiving comprises the steps:

a) supplying a plurality of objects, wherein an object for archiving has an object type and has at least one sub-object;
b) providing an assignment scheme that assigns a plurality of sub-object types to-the object type; and
c) identifying the at least one sub-object for archiving with the object, wherein the at least one sub-object has a sub-object type that is assigned to the object type of the object.

The first inventive computer-implemented method is complemented by a second inventive computer-implemented method on an archiving computer for archiving the objects that are identified in the first method. The second method comprises the steps:

a) receiving data for archiving, wherein the data comprise a first portion for an object having an object type and a second portion for at least one sub-object having a sub-object type. The sub-object is assigned to the object, wherein the assignment is identified by an assignment scheme that assigns the sub-object type to the object type, respectively.

b) archiving the data, wherein archiving means storing in a memory of the archiving computer.

A further preferred embodiment of the invention is implemented as first and second computer program products. The first computer program product has a plurality of instructions for causing a processor of the application computer to execute the steps of the first method to identify objects for archiving. The second computer program product has a plurality of instructions for causing a processor of the archiving computer to execute the steps of the second method to archive the objects that are identified by the first computer program product. The first and second computer program products can be stored on a first and second data carrier, respectively.

[0007] An advantage of the present invention is the combination of the enhanced flexibility in defining object types by composing subsets of sub-object types with the improved system control for assuring the completeness of archived objects. The first method always identifies all sub-objects for archiving that belong to an object on the application computer by collecting all sub-objects that have a sub-object type which is assigned to the object type of the object in the assignment scheme. If a new sub-object type gets assigned to the object type, the inventive first method automatically considers all sub-objects having the new sub-object type for archiving. The second method completes the archiving process.

[0008] A further advantage of the present invention is the independence of archiving programs from object/sub-object data structures. Therefore, the need for manual adjustments to archiving programs after the modification of the data structure of a corresponding object/sub-object type is eliminated. The first computer program product takes advantage of the object oriented nature of the documents. The first computer program product can handle any object having an object type where sub-object types are assigned to in the assignment scheme. Any addition, modification or removal of sub-objects in the assignment scheme is immediately considered by the first computer program product without any modification, because all sub-objects are identified on the base of sub-object type-to-object type assignments. The second computer program product completes the archiving process on the archiving computer.

[0009] The aspects of the invention will be realized and attained by means of the elements and combinations particularly pointed out in the appended claims. It is to be understood that both, the foregoing general description and the following detailed description are exemplary and explanatory only and are not restrictive of the invention as described.

Brief Description of the Drawings

[0010]

FIG. 1	illustrates a simplified block diagram of the inventive computer network system;
FIG. 2	illustrates a first method for identifying objects for archiving according to the present invention;
FIG. 3	illustrates a second method for archiving objects according to the present invention;
FIG. 4	illustrates objects having different object types;
FIG. 5	illustrates an assignment scheme with "sub-object type to object type" assignments;
FIG. 6	illustrates archiving data for an object; and
FIG. 7	illustrates an inventive application computer with an inventive archiving computer.

Detailed Description of the Invention

[0011] Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts. For convenience of explanation a list of references is provided prior to the claims.

[0012] FIG. 1 illustrates a simplified block diagram of the inventive computer network system 999 having a plurality of computers 900, 901, 902 (or 90q, with q=0...Q-1, Q any number).

[0013] Computers 900-902 are coupled via inter-computer network 990. Computer 900 comprises processor 910, memory 920, bus 930, and, optionally, input device 940 and output device 950 (I/O devices, user interface 960). As illustrated, the invention is present by computer program product 100 (CPP), program carrier 970 and program signal 980, collectively "program".

[0014] In respect to computer 900, computer 901/902 is sometimes referred to as "remote computer", computer 901/902 is, for example, a server, a router, a peer device or other common network node, and typically comprises many or all of the elements described relative to computer 900. Hence, elements 100 and 910-980 in computer 900 collectively illustrate also corresponding elements 10q and 91q-98q (shown for q=0) in computers 90q.

[0015] Computer 900 is, for example, a conventional personal computer (PC), a desktop and hand-held device, a multiprocessor computer, a pen computer, a microprocessor-based or programmable consumer electronics, a minicomputer, a mainframe computer, a personal mobile computing device, a mobile phone, a portable or stationary personal computer, a palmtop computer or the like.

[0016] Processor 910 is, for example, a central processing unit (CPU), a micro-controller unit (MCU), digital signal processor (DSP), or the like.

[0017] Memory 920 symbolizes elements that temporarily or permanently store data and instructions. Although memory 920 is conveniently illustrated as part of computer 900, memory function can also be implemented in network 990, in computers 901/902 and in processor 910 itself (e.g., cache, register), or elsewhere. Memory 920 can be a read only memory (ROM), a random access memory (RAM), or a memory with other access options. Memory 920 is physically implemented by computer-readable media, such as, for example: (a) magnetic media, like a hard disk, a floppy disk, or other magnetic disk, a tape, a cassette tape; (b) optical media, like optical disk (CD-ROM, digital versatile disk - DVD); (c) semiconductor media, like DRAM, SRAM, EPROM, EEPROM, memory stick, or by any other media, like paper.

[0018] Optionally, memory 920 is distributed across different media. Portions of memory 920 can be removable or non-removable. For reading from media and for writing in media, computer 900 uses devices well known in the art such as, for example, disk drives, tape drives.

[0019] Memory 920 stores support modules such as, for example, a basic input output system (BIOS), an operating system (OS), a program library, a compiler, an interpreter, and a text- processing tool. Support modules are commercially available and can be installed on computer 900 by those of skill in the art. For simplicity, these modules are not illustrated.

[0020] CPP 100 comprises program instructions and - optionally - data that cause processor 910 to execute method steps of the present invention. Method steps are explained with more detail below. In other words, CPP 100 defines the operation of computer 900 and its interaction in network system 999. For example and without the intention to be limiting, CPP 100 can be available as source code in any programming language, and as object code ("binary code") in a compiled form. Persons of skill in the art can use CPP 100 in connection with any of the above support modules (e.g., compiler, interpreter, operating system).

[0021] Although CPP 100 is illustrated as being stored in memory 920, CPP 100 can be located elsewhere. CPP 100 can also be embodied in carrier 970.

[0022] Carrier 970 is illustrated outside computer 900. For communicating CPP 100 to computer 900, carrier 970 is conveniently inserted into input device 940. Carrier 970 is implemented as any computer readable medium, such as a medium largely explained above (cf. memory 920). Generally, carrier 970 is an article of manufacture comprising a computer readable medium having computer readable program code means embodied therein for executing the method of the present invention. Further, program signal 980 can also embody computer program 100. Signal 980 travels on network 990 to computer 900.

[0023] Having described CPP 100, program carrier 970, and program signal 980 in connection with computer 900 is convenient. Optionally, program carrier 971/972 (not shown) and program signal 981/982 embody computer program product (CPP) 101/102 to be executed by processor 911/912 (not shown) in computers 901/902, respectively.

[0024] Input device 940 symbolizes a device that provides data and instructions for processing by computer 900. For example, device 940 is a keyboard, a pointing device (e.g., mouse, trackball, cursor direction keys), microphone, joystick, game pad, scanner. Although the examples are devices with human interaction, device 940 can also operate without human interaction, such as, a wireless receiver (e.g., with satellite dish or terrestrial antenna), a sensor (e.g., a thermometer), a counter (e.g., goods counter in a factory). Input device 940 can serve to read carrier 970.

[0025] Output device 950 symbolizes a device that presents instructions and data that have been processed. For example, a monitor or a display, (cathode ray tube (CRT), flat panel display, liquid crystal display (LCD), speaker, printer, plotter, vibration alert device. Similar as above, output device 950 communicates with the user, but it can also communicate with further computers.

[0026] Input device 940 and output device 950 can be combined to a single device; any device 940 and 950 can be provided optional.

[0027] Bus 930 and network 990 provide logical and physical connections by conveying instruction and data signals. While connections inside computer 900 are conveniently referred to as "bus 930", connections between computers 900-902 are referred to as "network 990". Optionally, network 990 comprises gateways being computers that specialize in data transmission and protocol conversion.

[0028] Devices 940 and 950 are coupled to computer 900 by bus 930 (as illustrated) or by network 990 (optional). While the signals inside computer 900 are mostly electrical signals, the signals in network are electrical, magnetic, optical or wireless (radio) signals.

[0029] Networking environments (as network 990) are commonplace in offices, enterprise-wide computer networks, intranets and the internet (i.e. world wide web). The physical distance between a remote computer and computer 900 is not important. Network 990 can be a wired or a wireless network. To name a few network implementations, network 990 is, for example, a local area network (LAN), a wide area network (WAN), a public switched telephone network (PSTN); a Integrated Services Digital Network (ISDN), an infra-red (IR) link, a radio link, like Universal Mobile Telecommunications System (UMTS), Global System for Mobile Communication (GSM), Code Division Multiple Access

(CDMA), or satellite link.

[0030] Transmission protocols and data formats are known, for example, as transmission control protocol/internet protocol (TCP/IP), hyper text transfer protocol (HTTP), secure HTTP, wireless application protocol, unique resource locator (URL), a unique resource identifier (URI), hyper text markup language HTML, extensible markup language (XML), extensible hyper text markup language (XHTML), wireless application markup language (WML), etc.

[0031] Interfaces coupled between the elements are also well known in the art. For simplicity, interfaces are not illustrated. An interface can be, for example, a serial port interface, a parallel port interface, a game port, a universal serial bus (USB) interface, an internal or external modem, a video adapter, or a sound card.

[0032] Computer and program are closely related. As used hereinafter, phrases, such as "the computer provides" and "the program provides", are convenient abbreviation to express actions by a computer that is controlled by a program.

FIG. 2 illustrates a first method 400 for identifying objects for archiving according to the present invention.

[0033] As used herein, objects are illustrated by icons with different shapes, wherein the shape of the icon (e.g. circle, hexagon, square) indicates the object type.

[0034] Method 400 comprises the steps

a) supplying 410 a plurality of objects 110, 111, 112 (cf. FIG. 4), wherein object 110 for archiving has object type 130 (cf. FIG. 5) and has at least one sub-object 120 (cf. FIG. 4);

b) providing 420 an assignment scheme 190 (cf. FIG. 5) that assigns a plurality of sub-object types 140, 141, 142 (cf. FIG. 5) to object type 130; and

c) identifying 430 the at least one sub-object 120 for archiving with object 110, wherein the at least one sub-object 120 has sub-object type 140 that is assigned to object type 130 of object 110.

Optionally (dashed lines), method 400 comprises the further step

d) archiving 440 the at least one sub-object 120 with object 110.

[0035] The method steps are now explained in detail.

[0036] In the supplying step 410, application computer 900 (cf. FIG. 7) provides the plurality of objects 110, 111, 112 that are subject to archiving. For example, an object for archiving is flagged with a corresponding archiving attribute. Typically, objects 110, 111, 112 are created by application programs (not shown) that run on application computer 900. In the example, object 110 has object type 130 (illustrated by a circle) and has at least one sub-object 120.

[0037] In the providing step 420, application computer 900 provides assignment scheme 190. Preferably, assignment scheme 190 is implemented as an assignment table (cf. table 3) in memory 920 (cf. FIG. 7) of application computer 900. A person of skill in the art can implement the assignment scheme in a different embodiment, such as an assignment program, at a different storage location in computer network system 999 (cf. FIG. 1). Assignment scheme 190 assigns a plurality of sub-object types 140, 141, 142 (cf. FIG. 5) to object type 130. As a consequence, object 110 having object type 130 can comprise sub-objects (e.g. 120) of any of the assigned sub-object types 140-142 (e.g. 140 illustrated as up-triangle).

Table 1:

assignment scheme 190	
Object Type	**Sub-Object Type**
130	140
130	141
130	142
...	...

[0038] In the identifying step 430, application computer 900 identifies all sub-objects (e.g. 120) of object 110 that are subject to archiving. This is achieved by a computer program that looks up all sub-object types 140, 141, 142 which are assigned to object type 130 of object 110 in assignment scheme 190. Then, the program automatically knows the data structure of object 110 and of all sub-objects (e.g.) 120 that can be assigned to object 110. The program identifies all sub-objects (e.g. 120) of object 110 that are subject to archiving by comparing the corresponding sub-object types to the sub-object types 140-142 in the assignment scheme. The advantage is that one computer program can be used for the identification of all sub-objects of any object

type, because the computer program learns, while executing, about the data structures of the sub-objects through the corresponding sub-object types.

[0039] In the optional archiving step 440 object 110 with all its sub-objects (e.g. 120) is stored. Preferably, object 110 and its sub-objects are stored in memory 921 (cf. FIG. 7) of archiving computer 901 (cf. FIG. 7). In case that the objects for archiving are identified and archived on the same computer, they can also be stored on application computer 900.

[0040] As pointed out in the summary of the invention a major advantage of the present invention is that the archiving of all sub-objects with a corresponding object is guaranteed. However, the present invention can also be used for applications, where partial archiving of an object is desired. In this case application scheme 190 is extended by an attribute that indicates sub-object types that are subject to archiving versus sub-object types not to be considered for archiving. Table 2 illustrates an example of extended assignment scheme 190. The "Archiving attribute column" comprises a flag that indicates that sub-object type 140 is a subject to archiving, whereas sub-object types 141, 142 are not.

Table 2:

extended assignment scheme 190		
Object Type	**Sub-Object Type**	**Archiving attribute**
130	140	yes
130	141	no
130	142	no
...	...	

FIG. 3 illustrates a second method 500 for archiving objects according to the present invention. Second method 500 complements first method 400 by storing the identified objects and sub-objects in archiving computer 901. Preferably, second method 500 is executed after identifying step 430 of method 400. Method 500 comprises the steps

a) receiving 510 data 160 (cf. FIG. 6) for archiving, wherein data 160 comprise a first portion 161 (cf. FIG. 6) for object 110 having object type 130 and a second portion 162 (cf. FIG. 6) for at least one sub-object 120 having sub-object type 140. Sub-object 120 is assigned to object 110, wherein the assignment is identified by using assignment scheme 190 that assigns sub-object type 140 to object type 130, respectively; and
b) archiving 520 data 160.

[0041] The steps are now explained in detail.

[0042] In the receiving step 510, preferably, archiving computer 901 receives data 160 from application computer 900 via network 990 (cf. FIG. 7). Data 160 comprise first portion 161 and second portion 162. First portion 161 includes object 110, which is subject to archiving and has object type 130. Second portion 162 includes all sub-objects of object 110 that are identified by application computer 900 according to the identifying step 430 of method 400 (cf. FIG. 2).

[0043] In the archiving step 520, preferably, archiving computer 901 stores data 160 in memory 921. Optionally, data 160 can be stored at any storage location of computer network system 999 (cf. FIG. 1).

FIG. 4 illustrates objects having different object types.

[0044] For example, objects 110, 111, 112 are subject to archiving. Each object has a different object type. The object type characterizes the different nature of the objects. For example, object 110 has object type 130 (cf. FIG. 5, circle) that can correspond to a document, such as a purchasing document, where items purchased from a certain vendor are listed. Object types of objects 111, 112 can represent documents, such as a financial document (e.g. an accounting voucher) or a sales document. Each object comprises at least one sub-object. For convenience of explanation, this is illustrated for object 110, only.

[0045] Object 110 comprises sub-object 120. Each sub-object has a defined sub-object type (e.g. business partner, note, product, etc.). In case of object 110 being a purchasing document, sub-object 120 having object type 140 (cf. FIG. 5), for example, refers to a line item that includes product data. An object can have multiple sub-objects of the same or of different object types.

FIG. 5 illustrates assignment scheme 190 with "sub-object type to object type" assignments 192-1 to 192-4.

[0046] An object type 130 comprises a specific subset of sub-object types 140-142. For example, a purchasing document object type comprises sub-object types for product data (e.g. 140 up-triangle), business partner (e.g. 141 down-triangle) and notes (e.g. 142 ellipse). Therefore, assignment scheme 190 defines assignments 192-1 to 192-3 of sub-object types 140-142 in column 191-2 to object type 130 in column 191-1. Further assignments 192-4 can be defined for any object type. However, an accounting voucher object type would not comprise a sub-object type for product data. New object types can be created by composing new subsets of sub-object types and assigning the new sub-sets to a new object type. The advantage is a high flexibility for data structures of objects (documents) through the combination of re-usable predefined data structures of sub-objects.

FIG. 6 illustrates archiving data 160 for object 110.

[0047] Data 160 include first portion 161 for object 110 and second portion 162 for all sub-objects (e.g. 120) that are archived together with object 110. The sub-objects in second portion 162 are identified according to the identifying step 430 (cf. FIG. 2). Preferably, data 160 comprise multiple data sets for all objects that are archived from application computer 900.

After having described the present invention as computer-implemented methods 400, 500, it will now be described as computer system.

FIG. 7 illustrates inventive application computer 900 with inventive archiving computer 901. In a preferred embodiment of the present invention, the two inventive computer systems 900, 901 are part of inventive computer network system 999 (cf. FIG. 1). For convenience of explanation, hardware components are shown with dashed lines and software components with solid lines.

[0048] Preferably, application computer 900 comprises memory 920 that supplies 410 (cf. FIG. 2) a plurality of objects 110, 111, 112 (cf. FIG 4) for archiving. Object 110 has object type 130 (cf. FIGS. 4,5) and has at least one sub-object 120 (cf. FIG 4). Memory 920 further provides 420 (cf. FIG. 2) assignment scheme 190 that assigns a plurality of sub-object types 140, 141, 142 (cf. FIG. 5) to object type 130.

[0049] Application computer 900 further comprises identifier 100-1 for identifying 430 (cf FIG. 2) the at least one sub-object 120 for archiving with object 110. Preferably, identifier 100-1 is implemented in computer program product 100 according to the identifying step 430 that is described in detail under FIG. 2.

[0050] Optionally, application computer 900 further comprises archiver 100-2. Preferably, archiver 100-2 is implemented in computer program product 100 according to the archiving step 440 that is described in detail under FIG. 2.

[0051] Preferably, inventive archiving computer 901 comprises interface 990-1 to receive 510 (cf. FIG. 3) data 160 from application computer 900 via network 990. Data 160 comprise first portion 161 and second portion 162. The purpose of first and second portions 161, 162 is explained in detail under FIG. 6. Preferably, archiving computer 901 further comprises memory 921 for archiving 520 data 160. Preferably, memory 921 is a database, such as a relational database, where data 160 are stored.

After having described the present invention as computer-implemented methods 400, 500 and computer systems 900, 901, it will now be described as computer program products 100/101 that can be stored on computer readable data carriers 970/971, respectively.

[0052] Preferably, first computer program product 100 (cf. FIG. 7) has a plurality of instructions for causing processor 910 of application computer 900 to identify objects for archiving. Computer program product 100 causes application computer 900 to execute the steps of method 400 (cf. FIG. 2).

[0053] Preferably, second computer program product 101 has a plurality of instructions for causing a processor 911 of archiving computer 901 (cf. FIG. 7) to archive objects. Second computer program product 101 causes archiving computer 901 to execute the steps of method 500 (cf. FIG. 3).

Reference	Description
100/101	computer program products
100-1	identifier
100-2	archiver
110-112	objects
120	sub-object
130	object type
140-142	sub-object types
160	archiving data
161, 162	portions of archiving data
190	assignment scheme
191-1, 191-2	columns of assignment scheme
192-1 to 192-4	assignments
400/500	methods
4xx/5xx	method steps

(continued)

Reference	Description
999	computer network system
900, 901, 902	computers
910, 911, 912	processors
920, 921, 922	memories
930	bus
940	input device
950	output device
960	user interface
970, 971	data carriers
980	program signal
990	network
990-1	interface

Reference numbers

Claims

1. A computer-implemented method (400) for identifying objects for archiving from a plurality of objects; the method (400) comprising the following steps:

   supplying (410) a plurality of objects (110, 111, 112) for archiving, wherein an object (110) has an object type (130) and has at least one sub-object (120);
   providing (420) an assignment scheme (190) that assigns a plurality of sub-object types (140, 141, 142) to the object type (130); and
   identifying (430) the at least one sub-object (120) for archiving with the object (110), wherein the at least one sub-object (120) has a sub-object type (140) that is assigned to the object type (130) of the object (110).

2. A method (500) for archiving objects; the method (500) comprising the steps:

   receiving (510) data (160) for archiving, wherein the data (160) comprise a first portion (161) for an object (110) having an object type (130) and a second portion (162) for at least one sub-object (120) having a sub-object type (140); the sub-object (120) being assigned to the object (110), wherein the assignment is identified by an assignment scheme (190) that assigns the sub-object type (140) to the object type (130), respectively; and
   archiving (520) the data (160).

3. The method (400) of claim 1, wherein in the supplying step (410) the object (110) is a document and the at least one sub-object (120) is a part of the document.

4. The method (400) of claim 1 comprising the further step
   archiving (440) the at least one sub-object (120) with the object (110).

5. A computer system (999) for archiving objects; the computer system comprising:

   an application computer (900) having a first memory (920); the first memory (920) supplying (410) a plurality of objects (110, 111, 112) for archiving, wherein an object (110) has an object type (130) and has at least one sub-object (120); the first memory (920) providing (420) an assignment scheme (190) that assigns a plurality of sub-object types (140, 141, 142) to the object type (130); the application computer (900) further having an identifier (100-1) for identifying (430) the at least one sub-object (120) for archiving with the object (110),

wherein the at least one sub-object (120) has a sub-object type (140) that is assigned to the object type (130) of the object (110); and
an archiving computer (901) having a second memory (921) for archiving (440) the at least one sub-object (120) with the object (110).

6. A computer system (900) for identifying objects for archiving; the computer system (900) comprising:

a memory (920) that supplies (410) a plurality of objects (110, 111, 112) for archiving, wherein an object (110) has an object type (130) and has at least one sub-object (120); the memory (920) providing (420) an assignment scheme (190) that assigns a plurality of sub-object types (140, 141, 142) to the object type (130); and an identifier (100-1) for identifying (430) the at least one sub-object (120) for archiving with the object (110), wherein the at least one sub-object (120) has a sub-object type (140) that is assigned to the object type (130) of the object (110).

7. The computer system (900) of claim 6 further comprising:

an archiver (100-2) for archiving (440) the at least one sub-object (120) with the object (110).

8. A computer system (901) for archiving objects; the computer system (901) comprising:

an interface (990-1) for receiving data (160) for archiving from a further computer (900), wherein the data (160) comprise a first portion (161) for an object (110) having an object type (130) and a second portion (162) for at least one sub-object (120) having a sub-object type (140); the sub-object (120) being assigned to the object (110), wherein the assignment is identified by an assignment scheme (190) that assigns the sub-object type (140) to the object type (130); and
a memory (921) for archiving (520) the data (160).

9. A computer program product (100) having a plurality of instructions for causing a processor (910) of a computer (900) to identify objects for archiving; the computer program product (100) causing the computer (900) to execute the following steps:

supplying (410) a plurality of objects (110, 111, 112) for archiving, wherein an object (110) has an object type (130) and has at least one sub-object (120);
providing (420) an assignment scheme (190) that assigns a plurality of sub-object types (140, 141, 142) to the object type (130); and
identifying (430) the at least one sub-object (120) for archiving with the object (110), wherein the at least one sub-object (120) has a sub-object type (140) that is assigned to the object type (130) of the object (110).

10. A computer program product (101) having a plurality of instructions for causing a processor (911) of a computer (901) to archive objects; the computer program product (101) causing the computer (901) to execute the following steps:

receiving (510) data (160) for archiving, wherein the data (160) comprise a first portion (161) for an object (110) having an object type (130) and a second portion (161) for at least one sub-object (120) having a sub-object type (140); the sub-object (120) being assigned to the object (110), wherein the assignment is identified by an assignment scheme (190) that assigns the sub-object type (140) to the object type (130), respectively.
archiving (520) the data (160).

11. A data carrier (970) readable by a computer (900); the data carrier (970) storing a plurality of instructions for causing a processor (910) of the computer (900) to identify objects for archiving; the plurality of instructions causing the computer (900) to execute the method (400) steps of claims 1 and 4.

12. A data carrier (971) readable by a computer (901); the data carrier (971) storing a plurality of instructions for causing a processor (911) of the computer (901) to archive objects; the plurality of instructions causing the computer (901) to execute the method (400) steps of claim 2.

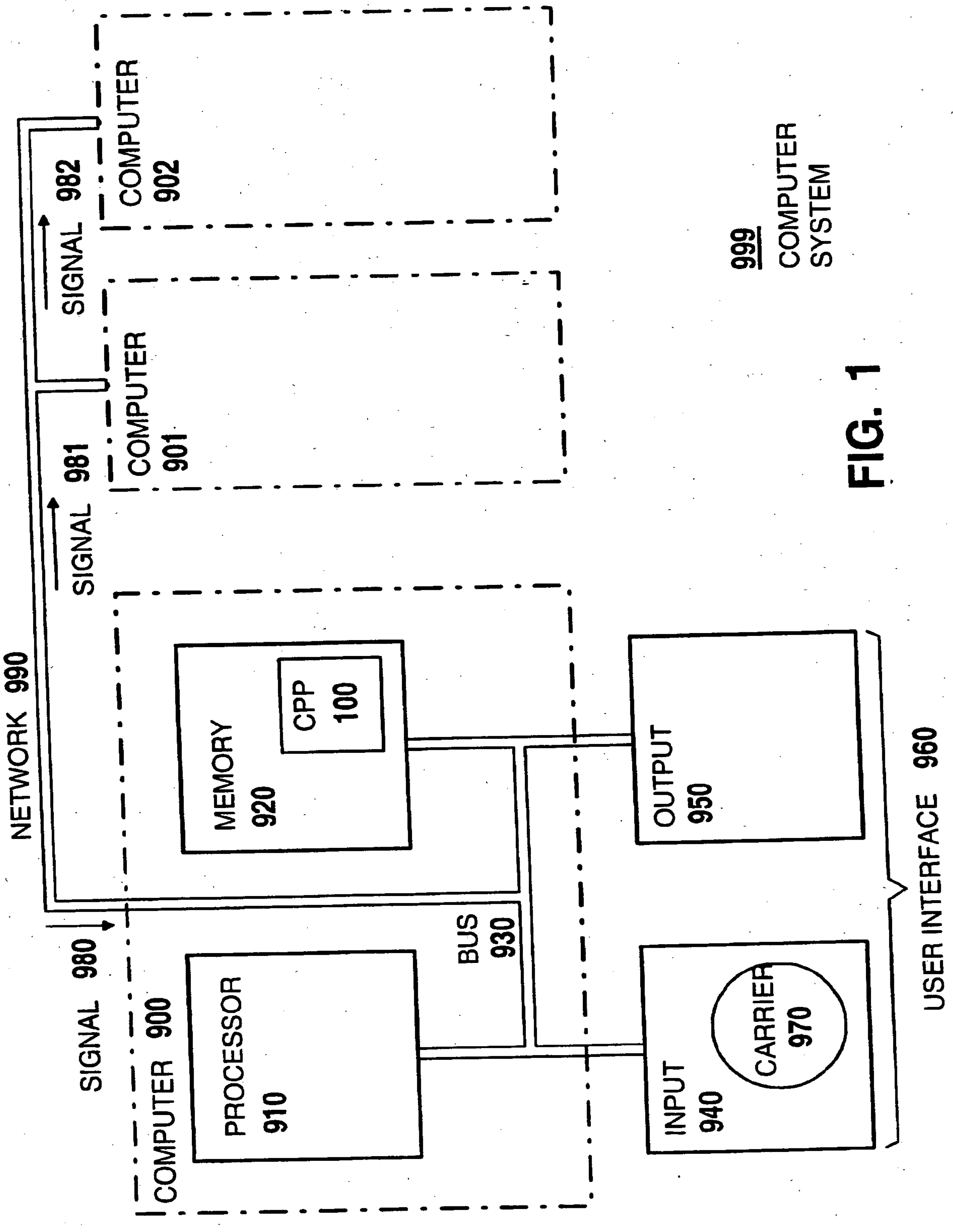
COMPUTER 900
PROCESSOR 910
MEMORY 920
CPP 100
BUS 930
INPUT 940
CARRIER 970
OUTPUT 950
USER INTERFACE 960
SIGNAL 980
NETWORK 990
SIGNAL 981
SIGNAL 982
COMPUTER 901
COMPUTER 902
999 COMPUTER SYSTEM

FIG. 1

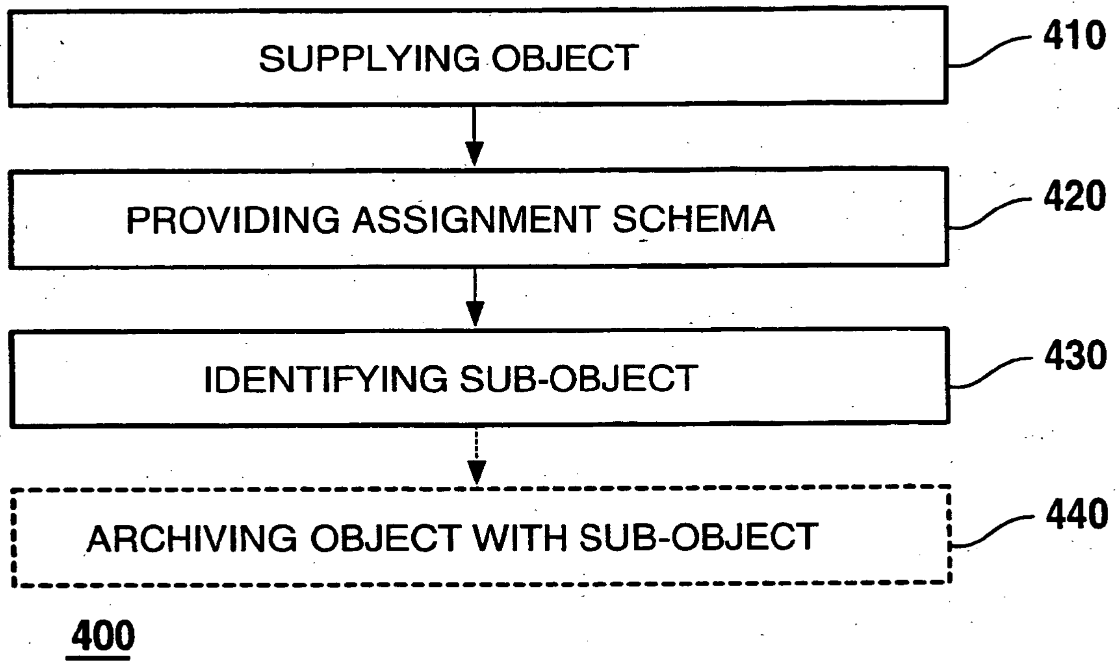
SUPPLYING OBJECT
410
PROVIDING ASSIGNMENT SCHEMA
420
IDENTIFYING SUB-OBJECT
430
ARCHIVING OBJECT WITH SUB-OBJECT
440
400

FIG. 2

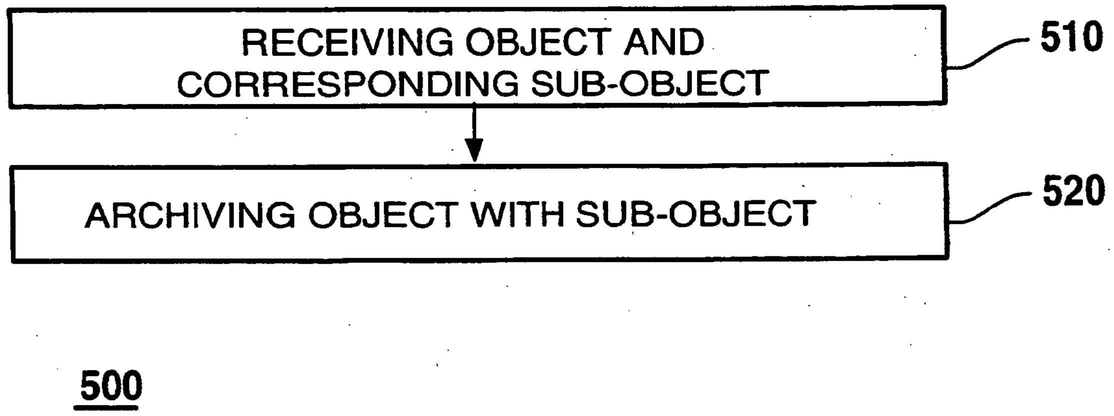
RECEIVING OBJECT AND CORRESPONDING SUB-OBJECT
510
ARCHIVING OBJECT WITH SUB-OBJECT
520
500

FIG. 3

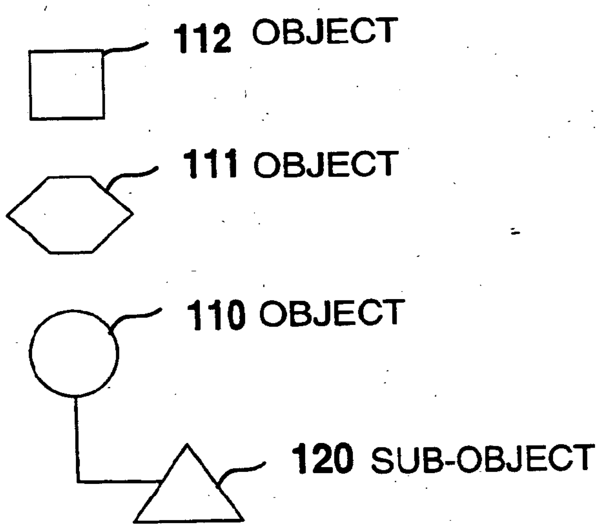
112 OBJECT
111 OBJECT
110 OBJECT
120 SUB-OBJECT

FIG. 4

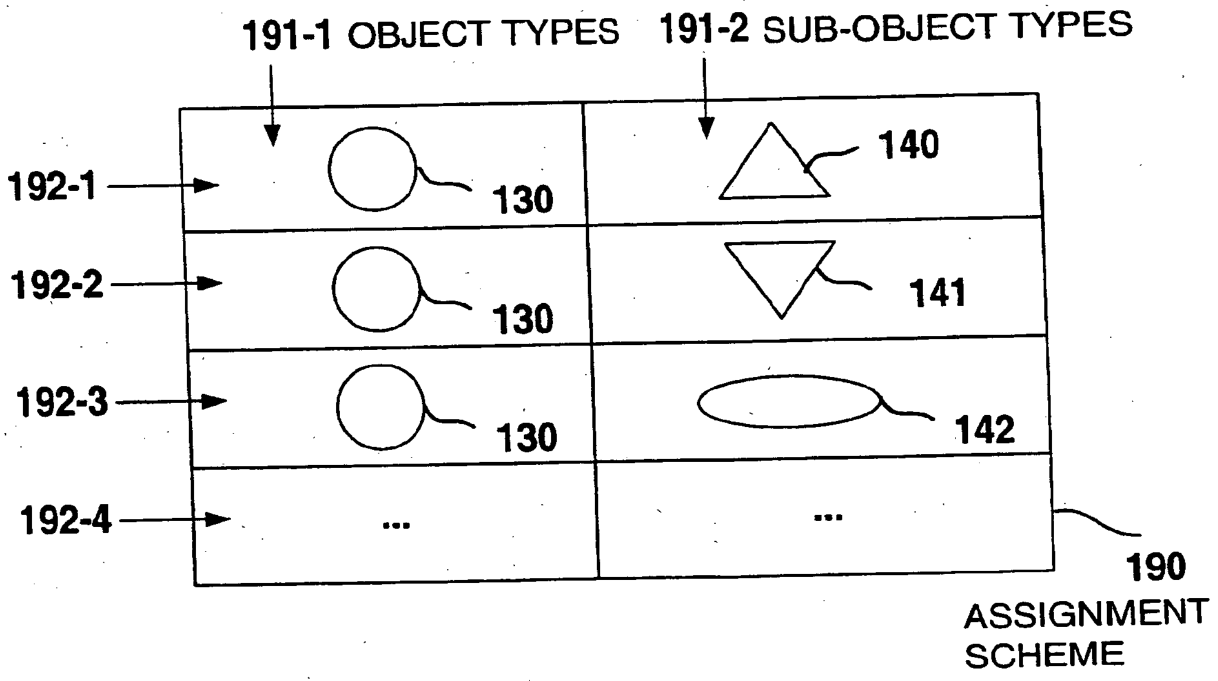
191-1 OBJECT TYPES
191-2 SUB-OBJECT TYPES
192-1
192-2
192-3
192-4
130
130
130
140
141
142
...
...
190
ASSIGNMENT SCHEME

FIG. 5

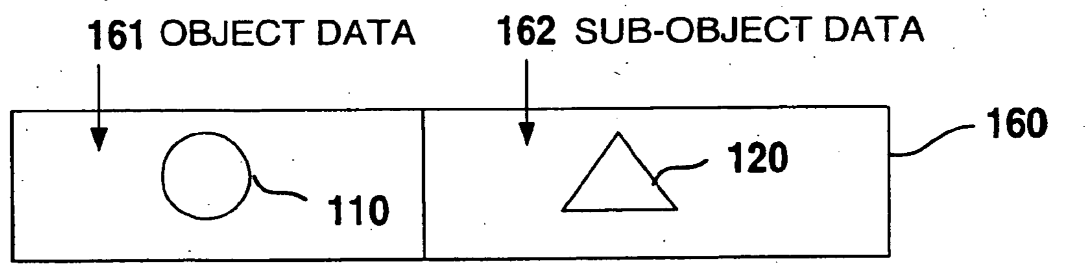
161 OBJECT DATA
162 SUB-OBJECT DATA
160
110
120

FIG. 6

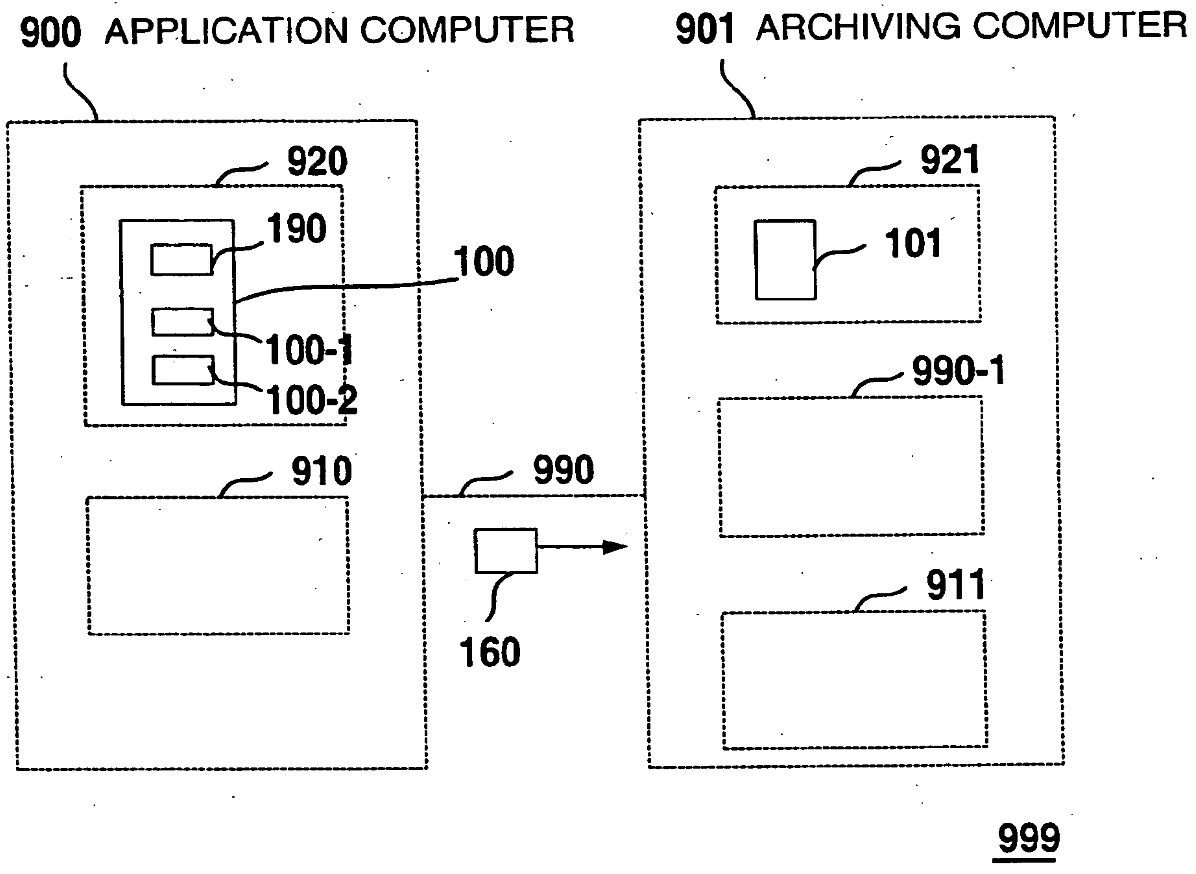
900 APPLICATION COMPUTER
901 ARCHIVING COMPUTER
920
190
100
100-1
100-2
910
990
160
921
101
990-1
911
999

FIG. 7

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 01 11 9078

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	US 5 875 441 A (NAKATSUYAMA HISASHI) 23 February 1999 (1999-02-23) * column 1, line 15 - line 50 * * column 7, line 40 - line 44 * * column 8, line 47 - line 65 * * column 3, line 22 - column 4, line 35 *	1-12	G06F17/30
A	HERBST A: "Long-term database support for EXPRESS data" SCIENTIFIC AND STATISTICAL DATABASE MANAGEMENT, 1994. PROCEEDINGS., SEVENTH INTERNATIONAL WORKING CONFERENCE ON CHARLOTTESVILLE, VA, USA 28-30 SEPT. 1994, LOS ALAMITOS, CA, USA,IEEE COMPUT. SOC, 28 September 1994 (1994-09-28), pages 207-216, XP010100561 ISBN: 0-8186-6610-2 * column 9 - column 10 *	1,2,5,6, 8-10	
A	US 5 581 755 A (NEUBAUER RONALD J ET AL) 3 December 1996 (1996-12-03) * column 3, line 57 - column 4, line 54 * * column 8, line 20 - column 8, line 46 *	1-12	TECHNICAL FIELDS SEARCHED (Int.Cl.7) G06F
A	US 6 236 988 B1 (ALDRED BARRY KEITH) 22 May 2001 (2001-05-22) * abstract * * column 1, line 22 - column 2, line 11 *	1-12	

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	14 June 2002	DE CASTRO PALOM.., L

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons
& : member of the same patent family, corresponding document

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 01 11 9078

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

14-06-2002

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5875441 A	23-02-1999	JP 9297768 A	18-11-1997
US 5581755 A	03-12-1996	NONE	
US 6236988 B1	22-05-2001	GB 2329044 A	10-03-1999

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82